REMARKS

Claim 7 stands objected to for a noted informality. Claims14-18 stand rejected under 35 USC §101 as being directed to non-statutory subject matter. Claim 1-18 stands rejected under 35 USC §103(a) as being unpatentable over Ziauddin et al., U.S. patent publication 2005/0177557 A1 in view of Ramasamy et al., U.S. patent 6,944,614.

Claims 1, 7, 8, 11, 14, and 15 have been amended to more clearly state the invention and to clearly distinguish over the references of record. Claims 2, 9, 10 and 16 have been cancelled. Reconsideration and allowance of each of the pending claims 1, 3-8, 11-15, and 17-18, as amended, is respectfully requested.

Claim 14 has been amended to more clearly state the invention and to more clearly recite statutory subject matter. Independent claim 14, as amended, recites a computer program product for implementing enhanced query governor functions in a computer system, said computer program product including instructions stored on a computer readable storage medium, wherein said instructions, when executed by the computer system to cause the computer system to perform the steps of. Thus, reconsideration and withdrawal of the rejection of claims 14-18 under 35 USC §101 as being directed to non-statutory subject matter, is respectfully requested.

Ziauddin et al., U.S. patent publication 2005/0177557 A1 discloses in the summary of the invention at paragraph [0008] that a run-away query execution is automatically identified by a background process that periodically looks at each of the currently executing queries and compares the current execution time with the execution

time estimated by the optimizer. Each query execution having a negative execution time difference can be automatically identified as a run-away query execution. The query execution plans that result in run-away executions can then be automatically tuned to produce more efficient execution plans. Paragraph [0007] states: Some vendors have used the idea of setting execution-time thresholds at various places in the execution plan to detect a case of run-away query execution. When a threshold is crossed during query execution, the run is aborted and the query sent back to the optimizer for re-optimization. But this method suffers from two drawbacks: setting of the thresholds and monitoring them at runtime incurs overhead, which can be significant and undesirable especially for light-weight queries, and the method of aborting a run and re-optimizing a query can be quite disruptive, especially if the run is aborted right before it was about to complete. Paragraph [0017] states: The automatic tuning optimizer (ATO), in a background process, then optimizes the execution plan for each query having a run-away execution by performing various analyses of the corresponding SQL statement, such as automatic identification and correction of inaccurate statistics, cardinality estimates, and cost estimates related to the statement, for example. If the execution plan built by the ATO is different from the one that is currently executing, the ATO can estimate how much more time the current plan execution is going to take to complete (remaining-time), as well as estimate how much time the new plan will take to execute (new-time). If the new-time is less than the remaining-time then the current plan run may be aborted and replaced with the new plan. Paragraph [0023] states: An example of a system 200 for automatic prevention

of run-away queries is shown in FIG. 2. A query optimizer, 210, receives a SQL statement, and generates an execution plan for the statement, which is executed by execution engine 220. An automatic performance monitor 230 identifies a potential run-away execution plan by observing the elapsed execution time of the plan, for example. The corresponding SQL statement is then input into an automatic tuning optimizer 240, which generates a profile 250 for the SQL statement. The profile can contain information related to missing or stale statistics. The profile can also include one or more tuning actions that can be used by an optimizer to generate an execution plan for the statement. The profile and the statement are received by the query optimizer 210, which generates a new execution plan, along with an estimated amount of time for executing the plan, based on the profile. The query optimizer also revises the estimated amount of time for executing the current plan using the profile. The time estimates are analyzed by a cost based plan selector, 260, which can determine that the current plan is a run-away plan if the corresponding execution time estimate is longer than that of the new plan. The plan selector 260 can also cause the execution engine 220 to abort the run-away plan and execute the new plan if the remaining amount of time to execute the run-away plan is more than the amount of time to execute the new plan. Otherwise, the execution engine continues to execute the current plan. In either case, query results 270 are returned by the system. Paragraph [0028] states: The Automatic Tuning Optimizer uses the past execution history of a SQL statement to determine the correct optimizer settings. For example, if the execution history shows that a SQL statement is only partially executed in the majority of times then the appropriate setting will be to

optimize it for first n rows, where n is derived from the execution history. This constitutes a customized parameter setting for the SQL statement. (Note that past execution statistics are available in the Automatic Workload Repository (AWR) presented later).

Ramasamy et al., U.S. patent 6,944,614 discloses a method, apparatus, article of manufacture, and a memory structure for monitoring an executed query comprising at least one execution thread. The method comprises the steps of executing the query; and while executing the query, storing an execution trace record for each execution thread in at least one execution log file. The execution trace record comprises execution trace information including a thread ID and a time stamp for the execution thread. The execution trace information can be recalled from the execution log file and presented to a user after execution of the query to allow post mortem analysis of the query. The apparatus comprises a data server for executing the execution thread and for storing an execution trace record for the executed execution thread, the execution trace record having execution trace information including a thread identifier and a time stamp; a query coordinator for storing an execution plan having a time stamp and for retrieving and synchronizing the execution trace record and the execution plan; and a client process for displaying the retrieved execution trace information to a user after execution of the query. FIGS. 3A-3C present an example of an SQL query, an associated tree of rational operators, and an associated access plan. FIG. 6 shows an example of an operator tree corresponding to the query shown in FIG. 3A. FIG. 7 presents an example of the operator tree depicted in FIG. 6 illustrating the

associated tree descriptors.

Reconsideration and allowance of each of the pending claims 1, 3-8, 11-15, and 17-18, as amended, is respectfully requested.

The invention enables the database user to be allowed to modify multiple query attributes including multiple executing components of a query. A query can be broken down into multiple query execution components, for example, data retrieval, trigger processing, and user defined function (UDF) processing, and with each of these query execution components having an individual time out value.

Independent claim 1, as amended, recites a method for implementing enhanced query governor functions comprising the steps of: monitoring events, responsive to an event to modify attributes, performing a modify attributes routine; said modify attributes routine including checking for a monitor being requested; and responsive to a monitor being requested, setting a timeout value for the requested monitor; the requested monitor including at least one of a user defined function (UDF) and a trigger; responsive to an event to execute query, performing an execute query routine; said execute query routine including: checking for a timeout value for the query and said at least one of said user defined function (UDF) and said trigger; responsive to identifying a timeout value for the query, resetting an execution time for the query; starting a monitor for each identified timeout value for the query and said at least one of said user defined function (UDF) and said trigger; starting the execution of the query; monitoring the execution of predefined events during the execution of the query; said predefined events including a begin or end of processing of said at least one of said trigger and said user defined

function (UDF); periodically checking execution status of the query; responsive to identifying the query is executing, checking for any expired timeout value; and halting the execution of the query responsive to an identified expired timeout value.

As amended, independent claim 1 further defines the modify attributes routine including checking for a monitor being requested; and responsive to a monitor being requested, setting a timeout value for the requested monitor; the requested monitor including at least one of a user defined function (UDF) and a trigger; responsive to an event to execute query, performing an execute query routine. As amended, independent claim 1 further defines the steps of the execute query routine to recite checking for a timeout value for the query and said at least one of said user defined function (UDF) and said trigger, responsive to identifying a timeout value for the query, resetting an execution time for the query; starting a monitor for each identified timeout value for the query and said at least one of said user defined function (UDF) and said trigger, and halting the execution of the query responsive to an identified expired timeout value. None of the cited references suggest checking for a monitor being requested; and responsive to a monitor being requested, setting a timeout value for the requested monitor; the requested monitor including at least one of a user defined function (UDF) and a trigger. None of the cited references suggest the steps of the execute query routine to recite checking for a timeout value for the query and said at least one of said user defined function (UDF) and said trigger, responsive to identifying a timeout value for the query, resetting an execution time for the query; starting a monitor for each identified timeout value for the query and said at least one of said user

defined function (UDF) and said trigger, and halting the execution of the query responsive to an identified expired timeout value. These limitations as recited in independent claim 1, as amended, are not disclosed or suggested by the combined teachings of Ziauddin et al. and Ramasamy et al.

Thus, independent claim 1, as amended, is patentable.

Independent claim 11, as amended, recites apparatus for implementing enhanced query governor functions comprising: a query governor program including a SQL processor program, said SQL processor program for monitoring events, and said SQL processor program responsive to an event to modify attributes, performing a modify attributes routine; and responsive to an event to execute query, performing an execute query routine; said modify attributes routine including the steps responsive to a monitor being requested, setting a timeout for the requested monitor; the requested monitor including at least one of a user defined function (UDF) monitor and a trigger monitor; said execute query routine including data retrieval processing, and at least one of user defined function (UDF) processing and trigger processing; and said query governor program including said at least one of said user defined function (UDF) monitor and said trigger monitor; said UDF monitor and said trigger monitor monitoring the execution of predefined events during the execution of the query; said predefined events including a begin or end of processing of said at least one of said trigger monitor and said UDF monitor. The reference of record including Ziauddin et al. and Ramasamy et al. do not teach or suggest a query governor program including a SQL processor program as now recited in independent claim 11, as amended. The claimed

SQL processor program for monitoring events, and said SQL processor program responsive to an event to modify attributes, performing a modify attributes routine; and responsive to an event to execute query, performing an execute query routine; said modify attributes routine including the steps responsive to a monitor being requested, setting a timeout for the requested monitor; the requested monitor including at least one of a user defined function (UDF) monitor and a trigger monitor is not disclosed or suggested in the prior art. Further the query governor program including said at least one of said user defined function (UDF) monitor and said trigger monitor; said UDF monitor and said trigger monitor monitoring the execution of predefined events during the execution of the query; said predefined events including a begin or end of processing of said at least one of said trigger monitor and said UDF monitor is not disclosed or suggested in the prior art.

Independent claim 14, as amended, recites a computer program product for implementing enhanced query governor functions in a computer system, said computer program product including instructions stored on a computer readable storage medium, wherein said instructions, when executed by the computer system to cause the computer system to perform the steps of: monitoring events, responsive to an event to modify attributes, performing a modify attributes routine; said modify attributes routine including checking for a monitor being requested; and responsive to a monitor being requested, setting a timeout value for the requested monitor; the requested monitor including at least one of a user defined function (UDF) and a trigger;

responsive to identifying an execute query event, performing an execute query

routine; said execute query routine including: checking for a timeout value for the query and said at least one of said user defined function (UDF) and said trigger, responsive to identifying a timeout value for the query, resetting an execution time for the query; starting a monitor for each identified timeout value for the query and said at least one of said user defined function (UDF) and said trigger; starting the execution of the query; monitoring the execution of predefined events during the execution of the query; said predefined events including a begin or end of processing of said at least one of said a trigger and said a user defined function (UDF); periodically checking execution status of the query; responsive to identifying the query is executing, checking for any expired timeout value; and halting the execution of the query responsive to an identified expired timeout value.

Thus, independent claim 11, as amended, is patentable.

Independent claim 14, as amended, is patentable for the same reasons as independent claim 1. None of the cited references suggest checking for a monitor being requested; and responsive to a monitor being requested, setting a timeout value for the requested monitor; the requested monitor including at least one of a user defined function (UDF) and a trigger. None of the cited references suggest the steps of the execute query routine to recite checking for a timeout value for the query and said at least one of said user defined function (UDF) and said trigger, responsive to identifying a timeout value for the query, resetting an execution time for the query; starting a monitor for each identified timeout value for the query and said at least one of said user defined function (UDF) and said trigger, and halting the execution of the query

responsive to an identified expired timeout value. These limitations as recited in independent claim 14, as amended, are not disclosed or suggested by the combined teachings of Ziauddin et al. and Ramasamy et al.

Thus, independent claim 14, as amended, is patentable.

Dependent claims 3-8, 12-13, 15, and 17-18 respectively depend from patentable claims 1, 11, and 14, further defining the invention. Each of the dependent claims 3-8, 12-13, 15, and 17-18, as amended, is likewise patentable.

Applicants have reviewed all the art of record, and respectfully submit that the claimed invention is patentable over all the art of record, including the references not relied upon by the Examiner for the rejection of the pending claims.

It is believed that the present application is now in condition for allowance and allowance of each of the pending claims 1, 3-8, 11-15, and 17-18, as amended, is respectfully requested. Prompt and favorable reconsideration is respectfully requested.

If the Examiner upon considering this amendment should find that a telephone interview would be helpful in expediting allowance of the present application, the Examiner is respectfully urged to call the applicants' attorney at the number listed below.

Respectfully submitted,

By: Joan Pennington

Joan Pennington
Reg. No. 30,885
Telephone: (312) 670-0736